Geometric Modeling and Algebraic Geometry

Bert Jüttler • Ragni Piene
Editors

Geometric Modeling and Algebraic Geometry

Springer

Bert Jüttler
Institute of Applied Geometry
Johannes Kepler University
Altenberger Str. 69
4040 Linz, Austria
bert.juettler@jku.at

Ragni Piene
CMA and Department of Mathematics
University of Oslo
P.O.Box 1053 Blindern
0136 Oslo, Norway
ragnip@math.uio.no

ISBN: 978-3-540-72184-0 e-ISBN: 978-3-540-72185-7

Library of Congress Control Number: 2007935446

Mathematics Subject Classification Numbers (2000): 65D17, 68U06, 53A05, 14P05, 14J26

Cover design: WMX Design GmbH, Heidelberg

Printed on acid-free paper

9 8 7 6 5 4 3 2 1

springer.com

Preface

The two fields of Geometric Modeling and Algebraic Geometry, though closely related, are traditionally represented by two almost disjoint scientific communities. Both fields deal with objects defined by algebraic equations, but the objects are studied in different ways. While algebraic geometry has developed impressive results for understanding the theoretical nature of these objects, geometric modeling focuses on practical applications of virtual shapes defined by algebraic equations. Recently, however, interaction between the two fields has stimulated new research. For instance, algorithms for solving intersection problems have benefited from contributions from the algebraic side.

The workshop series on Algebraic Geometry and Geometric Modeling (Vilnius 2002[1], Nice 2004[2]) and on Computational Methods for Algebraic Spline Surfaces (Kefermarkt 2003[3], Oslo 2005) have provided a forum for the interaction between the two fields. The present volume presents revised papers which have grown out of the 2005 Oslo workshop, which was aligned with the final review of the European project GAIA II, entitled *Intersection algorithms for geometry based IT-applications using approximate algebraic methods* (IST 2001-35512)[4].

It consists of 12 chapters, which are organized in 3 parts. The first part describes the aims and the results of the GAIA II project. Part 2 consists of 5 chapters covering results about special algebraic surfaces, such as Steiner surfaces, surfaces with many real singularities, monoid hypersurfaces, canal surfaces, and tensor-product surfaces of bidegree (1,2). The third part describes various algorithms for geometric computing. This includes chapters on parameterization, computation and analysis of ridges and umbilical points, surface-surface intersections, topology analysis and approximate implicitization.

[1] R. Goldman and R. Krasauskas, *Topics in Algebraic Geometry and Geometric Modeling*, Contemporary Mathematics, American Mathematical Society 2003.

[2] M. Elkadi, B. Mourrain and R. Piene, *Algebraic Geometry and Geometric Modeling*, Springer 2006.

[3] T. Dokken and B. Jüttler, *Computational Methods for Algebraic Spline Surfaces*, Springer 2005.

[4] `http://www.sintef.no/IST_GAIA`

The editors are indebted to the reviewers, whose comments have helped greatly to identify the manuscripts suitable for publication, and for improving many of them substantially. Thanks to Springer for the constructive cooperation during the production of this book. Special thanks go to Ms. Bayer for compiling the LaTeX sources into a single coherent manuscript.

Oslo and Linz,
August 2007

Bert Jüttler
Ragni Piene

Contents

Part I

Survey of the European project GAIA II

The European project GAIA II entitled *Intersection algorithms for geometry based IT-applications using approximate algebraic methods* (IST 2001-35512) involved six academic and industrial partners from five countries. The project aimed at combining knowledge from Computer Aided Geometric Design, classical algebraic geometry and real symbolic computation in order to improve intersection algorithms for Computer Aided Design systems. The project has has produced more than 50 scientific publications and several software toolkits, which are now partly available under the GNU GPL license.

We invited the coordinator of the project, Tor Dokken, to present a survey describing the background, the methods, the results and the achievements of the GAIA project. His summary is the first part of this volume.

1

The GAIA Project on Intersection and Implicitization

Tor Dokken

SINTEF ICT, Department of Mathematics
tor.dokken@sintef.no

Summary. In the GAIA-project we have combined knowledge from Computer Aided Geometric Design (CAGD), classical algebraic geometry and real symbolic computing to improve intersection algorithms for Computer Aided Design (CAD) systems. The focus has been on:

- Singular and near singular intersections of surfaces, where the surfaces are parallel or near parallel along segments of the intersection curves.
- Self-intersection of surfaces to facilitate trimming of self-intersecting surfaces.

The project has published more than 50 papers. Software toolkits from the project are available for downloading under the GNU GPL license.

1.1 Introduction

In the GAIA project we have combined knowledge from Computer Aided Geometric Design (CAGD), classical algebraic geometry and real symbolic computing to improve intersection algorithms for CAD-type systems. The calculation of the intersection between curves or surfaces can seem mathematically simple. This is true for the intersection of e.g. two straight lines when they intersect transversally and closed expressions for finding the intersection are used. However, if floating point arithmetic is used, care has to be taken to properly handle situations when the lines are parallel or near parallel. The intersection of two bi-cubic parametric surfaces can be reduced to finding the real zero set of a polynomial equation $f(s,t) = 0$ of bi-degree (54,54), which by itself is a challenging problem. In industrial systems floating point arithmetic is used, thus introducing rounding errors. In CAD system there are tolerances defining when two points are to be regarded as the same point. This has also to be taken into consideration in CAD-related intersection algorithms. The consequence of low quality intersection algorithms in CAD-systems is low quality CAD-models. Low quality CAD-models impose high costs on the product creation processes in industry.

The objectives of the GAIA project were related both to the scientific and technological aspects:

- To establish the theoretical foundation for a new generation of methods for intersection and self-intersection calculation of 3D CAD-type sculptured surfaces by introducing approximate algebraic methods and qualitative geometric descriptions.
- To demonstrate through software prototypes the feasibility of the approach.
- To investigate other uses of the approximate algebraic methods developed for extending functionality in modeling and interrogation of 3D geometries.
- To demonstrate that cooperation between mathematical domains such as approximation theory, classical algebraic geometry and computer aided geometric design is an important part of improving mathematical based technology on computers.
- To interact with CAD systems developers to improve both friendly use and robustness of future CAD systems.

To address these objectives the project activities have been structured in four main work areas, where each partner has had one or two work areas as their main focus:

- **Classification**, where we have used the tools and knowledge of classical algebraic geometry to better understand singularities, see Section 1.5.
- **Implicitization**, where we have looked into resultants and approximate implicitization to better find exact and approximate implicit representations of parametric surfaces, see Section 1.6.
- **Intersection**, where we have looked into algebraic intersection methods, combined numeric and algebraic intersection algorithms, and combined recursive and approximate implicit intersection methods, see Section 1.7.
- **Applications**, where we have searched for other problem domains where the approach of approximate implicitization can be used for better solving challenging problems related to systems of polynomial equations, see Section 1.8.

In addition to the topics above we will in this paper also address:

- Project background and partners in Section 1.2.
- Why CAD-type intersection is still a challenge in industry in Section 1.3.
- The algorithmic challenges of CAD type intersections in Section 1.4.
- The potential impact of the GAIA project in section 1.9.

The list of references at the end of this paper is a bibliography of papers related to the GAIA-project published by the project partners during and after the GAIA-project.

1.2 Project background and facts

The Ph.D. dissertation *Aspects of Intersection Algorithms* [16] from 1997 established close dialogue between the Department of Applied Mathematics at SINTEF ICT in Oslo, and the algebraic geometry group in the Department of Mathematics, University of Oslo. Gradually the idea of establishing a closer cooperation with other European groups matured, and the algebraic geometry group at the University of Nice

Sophia Antipolis in France was contacted. An application for an IST FET Open Assessment project was made also including the CAD-company think3. The proposal was successful, and in October 2000 the project *IST 1999-290010 – GAIA – Application of approximate algebraic geometry in industrial computer aided geometry* was started.

The final review of the assessment project in October 2001 was successful, and the project consortium was invited to propose a full FET-Open Project. Also this proposal was successful, and July 1st 2002 the project *IST-2001-35512 – GAIA II – Intersection algorithms for geometry based IT-applications using approximate algebraic methods* started. The full project ended on September 30th 2005. The budgets of the phases of project have been:

- GAIA assessment phase: Budget: 175 000 EURO, with financial contribution from the European Union of 100 000 EURO.
- GAIA II project phase: Budget: 2 300 000 EURO, with financial contribution from the European Union of 1 500 000 EURO.

Among the project partners we find one CAD-company, one industrial research institute, and four university groups:

- **SINTEF ICT, Department of Applied Mathematics, Norway**, has been the project coordinator, and focused on work within approximate implicitization, recursive intersection algorithms and recursive self-intersection algorithms. For more information on SINTEF see: http://www.sintef.no/math/.
- **think3 SPA, Italy and France,** is a CAD-system developer, and had as their main role to supply industrial level examples of challenging CAD-intersection and self-intersections, to integrate developed intersection algorithms into a prototype version of their system thinkdesign, and finally to test and assess the prototype algorithms developed in the project. For more information on think3 see: http://www.think3.com/.
- **University of Nice Sophia Antipolis (UNSA), France,** developed in close cooperation with INRIA exact intersection algorithms and a triangulation based reference method for surface intersection and self-intersection. For more information on UNSA and INRIA see: http://www-sop.inria.fr/galaad/.
- **University of Cantabria, Spain,** worked on combined numeric and exact intersection algorithms. For more information see: http://www.unican.es/.
- **Johannes Kepler University, Austria,** focused on new approaches to approximate implicitization and testing of approximate implicitization algorithms. For more information on this partner see: http://www.ag.jku.at/.
- **University of Oslo, Norway,** has focused on classification of algebraic curves and surfaces and their singularities. For more information on the University of Oslo see: http://www.cma.uio.no/.

Based on state-of-the-art reports, research reports and software prototypes we have tried to establish a common mathematical understanding of different approaches and tools. As the project partners come from an axis spanning from fairly theoretical classical algebraic geometry to computer aided geometric design and CAD-system developers, a major focus has been on bridging the language and knowledge gaps between the different mathematical groups involved. All groups have had to invest time into better understanding the traditional approaches of the other groups.

1.3 Why are CAD-type intersections still a problem for industry?

1.3.1 CAD technology evolution hampered by standardization

In the Workshop on Mathematical Foundations of CAD (Mathematical Sciences Research Institute, Berkeley, CA. June 4-5, 1999) the consensus was that: *The single greatest cause of poor reliability of CAD systems is lack of topologically consistent surface intersection algorithms.* Tom Peters, Computer Science and Engineering, The University of Connecticut, estimated the cost to be $1 Billion/year. For more information consult SIAM News, Volume 32, Number 5, June 1999, *Closing the Gap Between CAD Model and Downstream Application*, http://www.siam.org/siamnews/06-99/cadmodel.htm. Too low quality of CAD-intersection forces the industry to resort to expensive workarounds and redesigns to develop new products.

CAD-systems play a central role in most producing industries. The investment in CAD-model representation of current industrial products is enormous. CAD-models are important in all stages of the product life-cycle, some products have a short lifetime, while other products are expected to last at least for one decade. Consequently backward compatibility of CAD-systems with respect to functionality and the ability to handle "old" CAD-models is extremely important to the industry. Transfer of CAD-models between systems from different CAD-system vendors is essential to support a flexible product creation value chain. In the late 1980s the development of the **STEP standard** (ISO 10303) *Product Data Representation and Exchange* started with the aim to support backward compatibility of CAD-models and CAD-model exchange. STEP is now an important component in all CAD-systems and has been an important component in the globalization of design and production. However, STEP standardized the geometry processing technology of the 1980s, and the problems associated with that generation of technology. Due to the CAD-model legacy (the huge bulk of existing CAD-models) upgraded CAD-technology has to handle existing models to protect the resources already invested in CAD-models. Consequently the CAD-customers and CAD-vendors are conservative, and new technology has to be backward compliant. Improved intersection algorithms have thus to be compliant with STEP representation of geometry and the traditional approach to CAD coming from the late 1980s. For research within CAD-type intersection algorithms to be of interest to producing industries and CAD-vendors backward compatibility and the legacy of existing CAD-models have not to be forgotten.

1.4 Challenges of CAD-type intersections

If the faces of a CAD-represented volume are all planar, then it is fairly straightforward to represent the curves describing the edges with minimal rounding error.

However, if the faces are sculptured surfaces, e.g., bicubic NURBS - NonUniform Rational B-splines, the edges will in general be free form space curves with no simple closed mathematical description. As the tradition (and standard) within CAD is to represent such curves as NURBS curves, approximation of edge geometry with NURBS curves is necessary. For more information on the challenges of CAD-type intersections consult [54].

When designing within a CAD-system, *point equality tolerances* are defined that determine when two points should be regarded as the same. A typical value for such tolerances is 10^{-3}mm, however, some systems use tolerances as small as 10^{-6}mm. The smaller this tolerance is, the higher the quality of the CAD-model will be. Approximating the edge geometry with e.g., cubic spline interpolation that has fourth order convergence using a tolerance of 10^{-6} instead 10^{-3} will typically increase the amount of data necessary for representing the edge approximation by a factor between 5 and 6. Often the spatial extent of the CAD-models is around 1 meter. Using an approximation tolerance of 10^{-3}mm is thus an error of 10^{-6} relative to the spatial extent of the model.

The intersection functionality of a CAD-system must be able to recognise the topology of a model in the system. This implies that intersections between two faces that are limited by the same edge must be found. The complexity of finding an intersection depends on relative behaviour of the surfaces intersected along the intersection curve:

- **Transversal intersections** are intersection curves where the normals of the two surfaces intersected are well separated along the intersection curve. It is fairly simple to identify and localise the branches of the intersection when we only have transversal intersection.
- **Singular and near singular intersections** take place when the normals of the two surfaces intersected are parallel or near parallel in single points or along intervals of an intersection curve. In these cases the identification of the intersection branches is a major challenge.

Figures 1.1 and 1.2 respectively show transversal and near-singular intersection situations. In Figure 1.1 there is one unique intersection curve. The two surfaces in Figure 1.2 do not really intersect, there is a distance of 10^{-7} between the surfaces, but they are expected to be regarded as intersecting. To be able to find this curve, the point equality tolerance of the CAD-system must be considered. The intersection problem then becomes: Given two sculptured surface $f(u, v)$ and $g(s, t)$, find all points where $|f(u, v) - g(s, t)| < \varepsilon$ where ε is the point equality tolerance.

1.4.1 The algebraic complexity of intersections

The simplest example of an intersection of two curves in $\mathbb{R}^2$ is the intersection of two straight lines. Let two straight lines be given:

- A straight line represented as a parametric curve

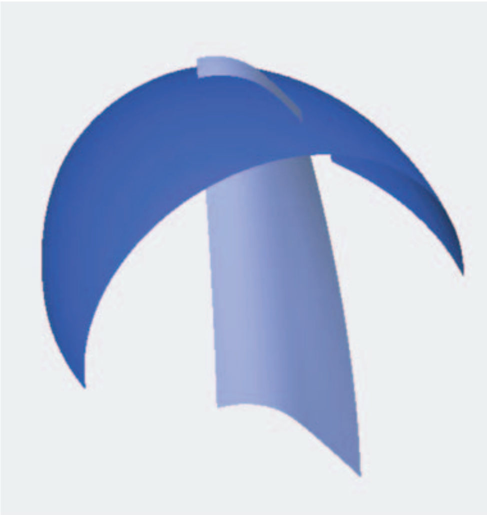

Fig. 1.1. Transversal intersection between two sculptured surfaces

Fig. 1.2. Tangential intersection between two surfaces

$$\mathbf{p}(t) = \mathbf{P}_0 + t\mathbf{T}_0, t \in \mathbb{R},$$

with $\mathbf{P}_0$ a point on the line and $\mathbf{T}_0$ the tangent direction of the line.

- A straight line represented as an implicit equation

$$q(x, y) = ((x, y) - \mathbf{P}_1) \cdot \mathbf{N}_1 = 0, (x, y) \in \mathbb{R}^2,$$

with $\mathbf{P}_1$ a point on the line, and $\mathbf{N}_1$ the normal of the line.

Combining the parametric and implicit representation the intersection is described by $q(\mathbf{p}(t)) = 0$, a linear equation in the variable t. Using exact arithmetic it is easy to classify the solution as:

- An empty set, if the lines are parallel.
- The whole line, if the lines coincide.
- One point, if lines are non-parallel.

Next we look at the intersection of two rational parametric curves of degree n and d, respectively. From algebraic geometry it is known that a rational parametric curve of degree d is contained in an implicit parametric curve of total degree d, see [27].

- The first curve is described as a rational parametric curve

$$\mathbf{p}(t) = \frac{\mathbf{p}_n t^n + \mathbf{p}_{d-1} t^{n-1} + \ldots + \mathbf{p}_0}{h_n t^n + h_{n-1} t^{n-1} + \ldots + h_0}.$$

- The second curve is described as an implicit curve of total degree d

$$q(x, y) = \sum_{i=0}^{d} \sum_{j=0}^{d-i} c_{i,j} x^i y^j = 0.$$

By combining the parametric and implicit representations, the intersection is described by $q(\mathbf{p}(t)) = 0$. This is a degree $n \times d$ equation in the variable t. As even the general quintic equation cannot be solved algebraically, a closed expression for the zeros of $q(\mathbf{p}(t))$ can in general only be given for $n \times d \le 4$. Thus, in general, the intersection of two rational cubic curves cannot be found as a closed expression. In CAD-systems we are not interested in the whole infinite curve, but only a bounded portion of the curve. So approaches and representations that can help us to limit the extent of the curves and the number of possible intersections will be advantageous.

We now turn to intersections of two surfaces. Let $\mathbf{p}(s, t)$ be a rational tensor product surface of bi-degree (n_1, n_2),

$$\mathbf{p}(s, t) = \frac{\sum_{i=0}^{n_1} \sum_{j=0}^{n_2} \mathbf{p}_{i,j} s^i t^j}{\sum_{i=0}^{n_1} \sum_{j=0}^{n_2} h_{i,j} s^i t^j}.$$

From algebraic geometry it is known that the implicit representation of $\mathbf{p}(s, t)$ has total algebraic degree $d = 2n_1 n_2$. The number of monomials in a polynomial of total degree d in 3 variables is $\binom{d+3}{3} = \frac{(d+1)(d+2)(d+3)}{6}$. So a bicubic rational surface has an implicit equation of total degree 18. This has 1330 monomials with corresponding coefficients.

Using this fact we can look at the complexity of the intersection of two rational bicubic surfaces $\mathbf{p}_1(u, v)$ and $\mathbf{p}_2(s, t)$. Assume that we know the implicit equation

$q_2(x, y, z) = 0$ of $\mathbf{p}_2(s, t)$. Combining the parametric description of $\mathbf{p}_1(u, v)$ and the implicit representation $q_2(x, y, z) = 0$ of $\mathbf{p}_2(s, t)$, we get $q_2(\mathbf{p}_1(u, v)) = 0$. This is a tensor product polynomial of bi-degree $(54, 54)$. The intersection of two bicubic patches is converted to finding the zero of

$$q_2(\mathbf{p}_1(u, v)) = \sum_{i=0}^{54}\sum_{j=0}^{54} c_{i,j} u^i v^j = 0.$$

This polynomial has $55 \times 55 = 3025$ monomials with corresponding coefficients, describing an algebraic curve of total degree 108. This illustrates that the intersection of seemingly simple surfaces can results in a very complex intersection topology. As in the case of curves, the surfaces we consider in CAGD are bounded, and we are interested in the solution only in a limited interval $(u, v) \in [a, b] \times [c, d]$.

1.5 Extend the use of algebraic geometry within CAD

The work within the GAIA project related to algebraic geometry and CAD has addressed three main topics:

- **Resultants** are one of the traditional methods for exact implicitization of rational parametric curves and surfaces. GAIA has produced some new results within this classical research area.
- **Singularities** in algebraic curves and surfaces are for understanding their geometry and topology.
- **Classification** is an old tradition in the field of Algebraic Geometry. It is a natural starting point when trying to understand the geometry of algebraic objects.

Papers on CAGD and algebraic methods from the project are [8, 9, 32, 33, 34, 35, 41, 42, 44, 48, 49, 57].

1.5.1 Resultants

The objective has been to develop tools for constructing, manipulating and exploiting implicit representations for parametric curves and surfaces based on resultant computations. The work in GAIA has been divided into three parts:

- A survey in four parts addressing:
  1. A resultant approach to detecting intersecting curves in P^3.
  2. Implicitizing rational hypersurfaces using approximation complexes.
  3. Using projection operators in Computer Aided Design.
  4. The method of moving surfaces for the implicitization of rational parametric surface in P^3.
- A report addressing sparse/toric resultant, results when the number of monomials is small compared to the number of possible monomials for polynomial of the degree in question.

- Development of prototypes of tools for constructing, manipulating and exploiting implicit representations for parametric curves and surfaces based on resultant computations.

One paper from the project addressing resultants is [7].

1.5.2 Singularities

Understanding the singularities of algebraic curves and surfaces is important for understanding the geometry of these curves and surfaces. A difficult problem in CAGD is the handling of self-intersections, and the theory of singularities of algebraic varieties is potentially a tool for handling this problem. In the GAIA project special emphasis has been put on detecting and locating singularities appearing on parameterized and implicitly given curves and surfaces of low degree.

- The presence of singularities affects the geometry of complex and real projective hypersurfaces and of their complements. We have illustrated the general principles and the main results by many explicit examples involving curves and surfaces.
- We have classified and analyzed the singularities of a surface patch given by a parameterization in order to proceed to an early detection. We distinguish algebraically defined surface patches and procedural surfaces given by evaluation of a program. Also we distinguish between singularities which can be detected by a local analysis of the parameterization and those which require a global analysis, and are more difficult to achieve.
- The detection of singularities is a critical ingredient of many geometrical problems, in particular in intersection operations. Once these critical points are located, one can for instance safely use numerical methods to follow curve branches. Detecting a singularity in a domain may also help in combining several types of methods.

A paper addressing singularities from the project is [48].

1.5.3 Classification

To use algebraic curves and surfaces in CAGD one needs to know about their shape: topology, singularities, self-intersections, etc. Most of this kind of classification theory is performed for algebraic curves and surfaces defined over the complex numbers, i.e., one considers complex (instead of only real) solutions to polynomial equations in two or three variables (or in three or four homogeneous variables, if the curves and surfaces are considered in projective space). Complete classification results exist only for low degree varieties (implicit curves and surfaces) and mostly only in the complex case. A simple example, the classification of conic sections, illustrates well that the classification over the real numbers is much more complicated than over the complex numbers.

We have collected known results about such classifications, especially concerning results for real curves and surfaces of low degree. Of particular interest in CAGD are parameterizable (i.e. so-called rational) curves and surfaces, and we have made explicit studies of various such objects. These objects, or patches of these objects, are potential candidates for approximate implicitization problems. For example, when the rough shape of a patch to be approximated is known, one can choose from a "catalogue" what kind of parameterized patch that is suitable - this eliminates many unknowns in the process of finding an equation for the approximating object and will therefore speed up the application. In addition to the survey of known results, particular objects that have been studied are:

- monoid curves and surfaces, especially quartic monoid surfaces
- tangent developables
- triangle and tensor surfaces of low degree of low (bi)degrees

Papers from the project addressing classification are [41, 42].

1.6 Exact and approximate implicitization

In CAD-type algorithms, combining parametric and algebraic representation of surfaces is in many algorithms advantageous. However, for surfaces of algebraic degree higher than two this is in general a very challenging task. E.g., a rational bi-cubic surface has algebraic degree 18. All rational surfaces have an algebraic representation. However, for surfaces of total degree higher than 3, not all algebraic surfaces will have a rational parametric representation. In the project we have the following two main approaches for change of representation.

1.6.1 Exact implicitization of rational parametric surfaces

General resultant techniques, but also specialized methods have been reviewed or developed in the GAIA II project to address the implicitization process:

- Projective, as well as anisotropic, resultants when the polynomials $f_0, \dots, f_3$ have no base points.
- Residual resultants when the polynomials have base points which are known and have special properties.
- Determinants of the so-called approximation complexes which give an implicit equation of the image of the polynomials as soon as the base points are locally defined by at most two equations.

Papers from the project addressing topics of exact implicitization are [6, 23, 24, 27, 47].

Approach	Comment	Addressed in GAIA II
Triangulation	Will both miss branches and produce false branches	See section 1.7.1 on the Reference Method
Lattice evaluation	Will miss branches	Used in many CAD-systems. Not addressed in GAIA II
Recursive	Guarantees topology within specified tolerances	See section 1.7.2 addressing the combination of recursion and approximate implicitization
Exact	Guarantees topology however will not always work	The AXEL library see Section 1.7.3
Combined exact & numeric	Guarantees topology however will not always work, faster than the exact methods	Uses Sturm Harbicht sequences for topology of algebraic curves, see Section 1.7.4

Table 1.1. Different CAD-intersection methods and their properties.

1.6.2 Approximate implicitization of rational parametric surfaces

Two main approaches have been pursued in the project.

- **Approximate implicitization by factorization** is a numerically stable method that reformulates implicitization to finding the smaller singular values of a matrix of real numbers. See one of [17, 21] for an introduction. The approach can be used as an exact implicitization method if the proper degree is chosen for the unknown implicit and exact arithmetic is used. The approach has high convergence rates and is numerical stable. Strategies for selecting solutions with a desired gradient behavior are supplied, either for encouraging vanishing gradients or avoiding vanishing gradients. The approach works both for rational parametric curves and surfaces, and for procedural surfaces. Experiments with piecewise algebraic curves and surfaces have produced implicit curves and surfaces that have more vanishing gradients than is desirable. We have experienced that estimating gradients will improve this situation. We have established a connection between the original approach to approximate implicitization, and a numerical integration based method that can also be used for procedural surfaces, and a sampling/interpolation based approach [22].
- **Approximate implicitization by point sampling and normal estimates** is constructive in nature as it estimates gradients of the implicit representation to ensure that gradients do not vanish when not desired [1, 2, 3, 11, 13, 36, 37, 38, 40, 50, 51, 52]. The approach produce good implicit curves and surfaces and the problem of vanish gradients in not desired regions is minimal. The method works well for approximation by piecewise implicit curves and surfaces.

The work within GAIA has illustrated the feasibility of approximate implicitization, established both new methods on approximate implicitization with respect to theory and practical use of approximate implicitization. It has also been important to compare the different approaches to approximate implicitization [59].

1.7 Intersection algorithms

In the GAIA II project phase the work on the reference method, see 1.7.1, continued from the assessment phase was completed. Further a completely new recursive intersection code has been developed addressing industrial CAD-type problems. Two more research oriented intersection codes have been developed: A pure symbolic code and a combined symbolic numeric code. See Table 1.1 for a short overview.

1.7.1 The reference method

The reference method is based on intersecting triangulations that approximate surfaces. This can be used for getting a fast impression of the possible existence of intersection or self-intersections. However, as the approach is sampling based, there is no guarantee that all intersections are found, the triangulations intersected can easily produce an incorrect topology of the intersection in near singular and singular cases, and even false intersection branches might be found. The development of the reference method has been important to allow think3 to develop the new user interfaces, and experiment with these before the software from the combined recursive and approximate implicit intersection code was available in its first versions.

1.7.2 Combined recursive and approximate implicitization intersection method

The combined recursive and approximate implicitization intersection was an extremely ambitious implementation task, the challenges of the implementation and approach is discussed in [20]. The ambition has been to address the very complex singular and near singular intersections. The aim was also Open Source distribution. Consequently a completely new intersection kernel had to be developed to ensure that we do not have any copyright problems. A major challenge with respect to self-intersections is the complexity of cusp curves intersecting self-intersection curves. The traditional approaches for recursive subdivision based intersection algorithms do not work properly in these cases. Thus when starting to test the code we entered unknown territory. By the end of GAIA II we could demonstrate that the approach works, but the stability of the toolkit was not at an industrial level. However, stabilization work on the code has continued after the GAIA II project.

Recursive intersection codes traditionally use Sinha's theorem that states that for a closed intersection loop to exist in the intersection of two surfaces then the normal fields of the surfaces have to overlap inside the loop. Consequently if there is no overlap of the normal fields of two surfaces they can not intersect in a closed loop. However, in singular intersections normal fields will overlap. In near singular intersections even deep levels of subdivision often do not separate the normal fields. In the GAIA II program code we have used approximate implicitization for separating the spatial extent of the surfaces, and for analyzing the possibilities of closed intersection loops by combining an approximate implicitization of one surface with the parametric representation of the other surface. This is a very efficient tool when

NURBS surfaces approximating low degree algebraic surfaces are intersected. Such approximating NURBS surfaces are frequently bi-cubic and are thus much more challenging to intersect that the algebraic surfaces they approximate. Approximate implicitization is used to find the approximate algebraic degree of the surfaces, and consequently simplifies the intersection problem significantly.

The high-level reference documentation of the software has already been produced in doxygen and is available on the web. Other papers on numeric intersection algorithms from the project are [5, 14, 54, 55, 56].

1.7.3 Algebraic methods

The problems encountered in CAGD are sometimes reminiscent of 19th century problems. At that time, realizing the difficulties one had working in affine instead of projective space, and over the real numbers instead of the complex numbers one soon shifted the theoretical work towards projective geometry over the complex numbers. In fact, it is still in this situation that the modern intersection theory from algebraic geometry works best:

- **Bisection through a Multidimensional Sturm Theorem.** A variant of the classical Sturm sequence is presented for computing the number of real solutions of a polynomial system of equations inside a prescribed box. The advantage of this technique is based on the possibility of being used to derive bisection algorithms towards the isolation of the searched real solutions.
- **Algorithms for exact intersection.** Algorithms using Sturm–Habicht based methods have been implemented and are available at Axel - Algebraic Software Components for gEometric modeLing.

Papers on exact intersection methods from the project are [15, 30].

1.7.4 Combined algebraic numeric methods

The approach for the combined methods is to combine the rational parametric description of one surface $\mathbf{p}_1(s,t)$, with the algebraic representation of the other surface $q_2(x,y,z) = 0$. Thus the problem is converted to a problem of finding the topology of an algebraic curve $q_2(\mathbf{p}_1(s,t)) = 0$ in the parameterization of the first surface:

- **A limited number of critical points.** The approach is based on finding critical point, points where either $\nabla f(s,t) = 0$ or $\partial f(s,t)/\partial s = 0$. For any value in the first parameter direction of $f(s,t)$ there will be a limited number of such critical points. There is also a finite number of rotations of $f(s,t)$ that will have more than one critical point. $f(s,t)$ is rotated to ensure that for a given value there will be only one critical point.
- **Projection to first parameter direction.** The problem is project to a polynomial in the first parameter variable of $f(s,t)$ by computing the discriminant $R(s)$ of $f(s,t)$ with respect to t, and finding the real root of $R(s)$, $\alpha_1,\ldots,\alpha_r$.. The

Sturm-Habicht sequences here supply an exact number of real roots in the interval of interest.

- **Finding values in the second parameter direction.**Then for each α_i $i = 1, \ldots, r$ we compute the real roots of $f(\alpha_i, t)$, $\beta_{i,j}$, $j = 1, \ldots, s_i$,. For every α_i and $\beta_{i,j}$ compute the number of half branches to the right and left of the point $(\alpha_i, \beta_{i,j})$.
- **Reconstruction of topology of the algebraic curve.** From the above information the topology of the algebraic curve in the domain of interest can be constructed.

Papers on this approach in the project are [4, 10, 28, 29, 31].

To ensure the approach to work the root computation has to use extended precision to ensure that we reproduce the number of roots predicted by the Sturm-Habicht sequences. The algorithms have been developed using symbolic packages.

1.8 New applications of the approach of approximate implicitization

A number of different applications of approximated implicitization are addressed in the subsections following.

1.8.1 Closest point foot point calculations

Inspired by approximate implicitization this problem has been addressed by modeling moving surfaces normal to the surface and intersecting in constant parameter lines [57]. The set up of the problems follows the ideas of approximate implicitization; singular value decomposition is used to find the coefficients of the moving surfaces. By inserting the coordinates of a point into such a moving surface a polynomial equation in one variable results. The zeros of this identify constant parameter lines with a foot point. Further a theory addressing the algebraic and parametric degree of the moving surface is established.

1.8.2 Constraint solving

Multiple constraints described by parametric curves, surfaces or hypersurfaces over a domain used for optimization can be modeled using approximate implicitization as a piecewise algebraic curve, or surface, or hypersurface. Thus a very compact way of modeling constraints has been identified.

1.8.3 Robotics

Within robotics we have identified a number of uses. We have experimented with checking for self-intersection of robot tracks. CAD-surfaces used in robot planning can check for self-intersections by the GAIA tools. The control of advanced robots can be expressed as systems of polynomial equations. To solve such equations the